**RELATÓRIO DE PROJETO**
**versão 2.1**

# Manual de instruções

## 1.Termo de Referência

Esse relatório diz respeito ao edital **número 46**, "**OBJ-REL - Camada de Persistência Objeto-Relacional"**, publicado entre os dias 04 a 11 de maio de 2008, a ser realizado no período de maio/2008 a outubro/2008.

## 2.Introdução

De acordo com o descrito no termo de referência desse projeto, os frameworks utilizados pelo Interlegis permitem o armazenamento de informações tanto em banco de dados relacionais (PostgreSQL e MySQL) quanto em banco de dados orientados a objetos (ZODB).

De forma geral, quando trata-se de representar informações dentro do Plone (utilizando-se o framework Archetypes), a persistência dos dados costuma ser orientada a objetos (OO) e realizada diretamente no banco de dados orientado a objetos ZODB. Há algumas soluções do Interlegis como, por exemplo, o SAPL (Sistema de Apoio ao Processo Legislativo), que podem ser consideradas soluções "híbridas", armazenando informações em diferentes modelos (parte das informações em banco de dados relacional e parte em banco de dados orientado a objetos). No entanto, o SAPL não utiliza o framework Archetypes e, por sua vez, não está relacionado com esse projeto.

Esse relatório tratará apenas das considerações relevantes ao primeiro caso, o mapeamento de informações objeto-relacional. Nesse caso, as informações que normalmente seriam armazenadas em bancos de dados OO precisam ser mapeadas para um banco de dados relacional.

Esse terceiro relatório complementa os Relatório 1 e 2 já entregues anteriormente e diz respeito a entrega do "manual de instruções". Posteriormente, o relatório 4 tratará a respeito última entrega, referente aos procedimentos de transferência da tecnologia para o Interlegis.

## 3. Manual de Instruções

Como descrito no Relatório 2, a solução de mapeamento objeto-relacional implementada recebeu o nome de **interlegis.sqlalchemystorage** e tem como dependência principal o módulo **archetypes.sqlalchemystorage**. A instalação da solução completa, assim como sua configuração, não é um processo trivial.

Nesse sentido, tentou-se seguir as boas práticas de empacotamento de módulos Python, usando buildout e virtualenv, para que o processo fosse simplificado. O processo de instalação é descrito a seguir detalhadamente.

## 4. Instalação

O primeiro passo é verificar a versão instalada do interpretador python. A instalação do interpretador **python versão 2.4.4** deve ter sido feita a partir de pacotes ou, quando feita manualmente, deve incluir o suporte a biblioteca "zlib" (e, para isso, basta que no momento da compilação esteja instalado o pacote zlib-dev ou seu equivalente).

Para realizar a instalação a partir de pacotes ".deb", podemos utilizar o comando a seguir (por convensão, os comandos precedidos por **#** devem ser executados como root ou através do sudo e os comandos precedidos por **$** podem ser executados por um usuário comum):

```
# apt-get install python2.4 python2.4-dev python2.4-imaging python2.4-mysqldb
build-essential
```

Os pacotes python2.4-imaging e python2.4-mysqldb também podem ser instalados via buildout, mas constatou-se que esse procedimento automático falha em algumas versões da distribuição Ubuntu e que pode ser útil manter as linhas referentes a instalação do PIL e do MySQLdb comentadas no buildout e é aparentemente mais simples utilizar os pacotes padrão da distribuição. O pacote build-essential, por sua vez, é necessário para a compilação do Zope (que possui partes de seu código em C e precisa do compilador gcc e do make).

Para testar se o python2.4 está corretamente instalado, assim como o python2.4-imaging e python2.4-mysqldb, invocar o interpretador e tentar importar os módulos correspondentes:

```
$ python2.4
import PIL
import MySQLdb
```

Com o interpretador python versão 2.4 instalado, devemos instalar o easy install e o virtualenv. Para isso, recomenda-se a utilização dos seguintes comandos:

```
# wget http://peak.telecommunity.com/dist/ez_setup.py
# python2.4 ez_setup.py
# easy_install-2.4 virtualenv
```

Caso o python2.4 ou o easy_install-2.4 não estejam contidos na variável de ambiente PATH, convém adicioná-los (isso geralmente é necessário apenas quando o interpretador python foi instalado manualmente). Isso pode ser feito adicionando a seguinte linha no final do arquivo /etc/profile

```
export PATH=/PATH_DE_INSTALACAO_DO_PYTHON/bin/:$PATH
```

O `PATH_DE_INSTALACAO_DO_PYTHON` costuma ser o `/usr/local`.

Para prosseguirmos, é preciso criar o diretório base da instalação. Esse processo é realizado com o apoio de um "checkout" do buildout, feito a partir de um repositório SVN, seguido da criação de um ambiente virtual "virtualenv", da instalação do "ZopeSkel" e da execução do "buildout". Assim, recomenda-se a utilização dos seguintes comandos:

```
$ svn co https://dev.pytown.com/svn/interlegis.sqlalchemystorage/buildout/
interlegis-plone25
$ virtualenv interlegis-plone25
$ cd interlegis-plone25
$ ./bin/python bootstrap.py
$ ./bin/easy_install ZopeSkel
$ ./bin/buildout -vv
$ mkdir -p parts/instance/var/fss_storage parts/instance/var/fss_backup
```

Para carregar o código fonte a partir do repositório SVN via EXTERNALS.txt, utilizamos os comandos a seguir.

```
$ cd src; svn up; cd ..
```

O arquivo src/EXTERNALS.txt define a partir de qual repositório o framework interlegis.sqlalchemystorage e sua dependência principal, o módulo archetypes.sqlalchemystorage, são atualizados. Quando houver qualquer correção ou melhoria implementada, a atualização consistirá apenas em repetir os comandos acima e reiniciar a instancia do Zope.

```
#
# Create this file: svn propget svn:externals . > EXTERNALS.txt
# Apply this file: svn propset svn:externals -F EXTERNALS.txt .
#
archetypes.sqlalchemystorage https://svn.plone.org/svn/archetypes/
archetypes.sqlalchemystorage/branches/weimar-improvements
interlegis.sqlalchemystorage https://dev.pytown.com/svn/
interlegis.sqlalchemystorage/trunk
```

Percebe-se no arquivo de configuração acima que parte do “buildout” desenvolvido referente ao módulo interlegis.sqlalchemystorage está armazenada em um repositório SVN privado (dev.pytown.com), mas que o módulo archetypes.sqlalchemystorage está armazenado no repositório oficial do framework Archetypes (svn.plone.org).

Durante o desenvolvimento do projeto ficou claro que seriam necessárias diversas mudanças no módulo archetypes.sqlalchemystorage e, por essa razão, as mesmas foram feiras em um branch específico. O repositório svn.plone.org é público, mas com acesso de “escrita” restrito. Por outro lado, o repositório dev.pytown.com tem acesso restrito tanto para escrita quanto para leitura, mas **cabe ao Interlegis decidir onde será mantido o código fonte produzido**; o repositório em questão foi utilizado para simples conveniência. Cabe ao Interlegis definir se os produtos desenvolvidos serão mantidos nos servidores do próprio interlegis ou mesmo liberados publicamente no repositório “collective” do Plone.

## 5.Configuração do MySQL

Antes de iniciar o Zope é preciso iniciar o servidor MySQL, criando-se os bancos de dados de produção e teste, com as devidas permissões. Cada Plone site pode ter uma conexão para um banco de dados diferente. Existe também uma definição de conexão padrão usada pelos testes automatizados (descritos como doctests).

Por convensão, adotou-se que os bancos de dados de testes e de produção usam respectivamente as seguintes strings de conexão:

- Desenvolvimento: "mysql://interlegis:interlegis@localhost/testinterlegis"
- Produção: "mysql://interlegis:interlegis@localhost/ID_PLONE_SITE"

Em produção, cada Plone Site pode ter um banco de dados diferente, onde o nome do banco de dados é o id do objeto Plone Site. Essa string de conexão padrão pode ser alterada em uma property na ferramenta portal_properties/site_properties. Assim, se considerarmos que o id de um Plone Site em produção seria “interlegis”, temos:

```
$ mysql -u root -p
Enter password:

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.0.51a-3ubuntu5.1 (Ubuntu)
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database interlegis;
Query OK, 1 row affected (0.12 sec)

mysql> create database testinterlegis;
Query OK, 1 row affected (0.00 sec)

mysql> grant all on interlegis.* to interlegis@localhost identified by
'interlegis';
Query OK, 0 rows affected (0.00 sec)

mysql> grant all on testinterlegis.* to interlegis@localhost identified by
'interlegis';
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
```

Após ajustar as permissões do banco de dados, é importante verificar se o engine padrão de persistência do MySQL é o InnoDB. Esse engine permite a criação de

tabelas com suporte transacional, com o recurso de "rollback". Para isso, recomenda-se que o arquivo /etc/mysql/my.cnf contenha:

```
[mysqld]
...
default-storage_engine = innodb
```

## 6.Configuração do Zope

Após configurar as permissões e o engine padrão de persistência no servidor MySQL, podemos testar a inicialização do servidor Zope. Para isso, podemos iniciar o Zope em modo "foreground" com o seguinte comando:

```
$ ./bin/instance fg
```

Se durante a primeira tentativa de inicializar a instancia do Zope ocorrer o seguinte erro:

```
/opt/zope/interlegis-plone25/parts/instance/bin/runzope -X debug-mode=on
2008-10-15 12:50:58 INFO ZServer HTTP server started at Wed Oct 15 12:50:58 2008
       Hostname: 0.0.0.0
       Port: 8090
2008-10-15 12:50:58 CRITICAL Zope A user was not specified to setuid to; fix this to start as
root (change the effective-user directive in zope.conf)
Traceback (most recent call last):
  File "/opt/zope/interlegis-plone25/parts/zope2/lib/python/Zope2/Startup/run.py", line 56,
in ?
    run()
  File "/opt/zope/interlegis-plone25/parts/zope2/lib/python/Zope2/Startup/run.py", line 21,
in run
    starter.prepare()
  File "/opt/zope/interlegis-plone25/parts/zope2/lib/python/Zope2/Startup/__init__.py", line
94, in prepare
    self.dropPrivileges()
  File "/opt/zope/interlegis-plone25/parts/zope2/lib/python/Zope2/Startup/__init__.py", line
213, in dropPrivileges
    return dropPrivileges(self.cfg)
  File "/opt/zope/interlegis-plone25/parts/zope2/lib/python/Zope2/Startup/__init__.py", line
382, in dropPrivileges
    raise ZConfig.ConfigurationError(msg)
ZConfig.ConfigurationError: A user was not specified to setuid to; fix this to start as root
(change the effective-user directive in zope.conf)
```

significa que a instancia não pode ser inicializada com o usuário (geralmente o usuário root ou outro usuário que não tenha permissão de escrita no diretório parts/instance/var). Para corrigir isso, pode-se alterar a diretiva effective-user do etc/zope.conf ou efetuar ajustes nas permissões usando o comando chmod.

## 7.Testes Automatizados

Para executar os testes automatizados que cobrem a implementação, interrompe-se a execução do Zope (pressionando-se ctrl-c caso ele esteja rodando em modo foreground ou usando o comando "`$ ./bin/instance stop`" caso ele esteja rodando em modo background) e em seguida edita-se o arquivo de configuração config.py, como demonstrado a seguir:

```
$ vim src/interlegis.sqlalchemystorage/interlegis/sqlalchemystorage/config.py
TEST_MODE = True
CLEAN_DB = True
```

E então executa-se os testes automatizados:

```
./bin/instance test -p "interlegis.sqlalchemystorage"
```

Durante a execução dos testes não deve ser reportada nenhuma falha ou erro. Se isso ocorrer, é possível que o suporte ao engine InnoDB do MySQL não esteja funcionando e, por sua vez, a integridade transacional entre o Zope e o MySQL esteja sendo afetada.

Para voltar a executar o Zope usando o banco de dados de produção, deve-se alterar novamente o arquivo config.py para:

```
TEST_MODE = False
CLEAN_DB = False
```

## 8.Rodando o Zope

Depois de rodar os testes automatizados e verificar que tudo funciona como esperado, podemos inicializar o servidor Zope (com o comando "`$ ./bin/instance stop`"), abrir um navegador e acessar o seguinte endereço:

```
url: http://IP_SERVIDOR:8090/manage
usuario: admin
senha: admin
```

A porta 8090 assim como o usuário e senha "admin" são definidos como padrão pelo buildout.